Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

# Interpretable Temporal Attention Network for COVID-19 forecasting

Binggui Zhou [a,b], Guanghua Yang [a,*], Zheng Shi [a,b], Shaodan Ma [b]

[a] School of Intelligent Systems Science and Engineering, Jinan University, Zhuhai 519070, China
[b] State Key Laboratory of Internet of Things for Smart City and Department of Electrical and Computer Engineering, University of Macau, 999078, Macao Special Administrative Region of China

ARTICLE INFO

*Article history:*
Received 30 October 2021
Received in revised form 17 February 2022
Accepted 26 February 2022
Available online 9 March 2022

*Keywords:*
COVID-19 forecasting
Neural network
Covariate forecasting
Multi-task learning
Degraded Teacher Forcing

ABSTRACT

The worldwide outbreak of coronavirus disease 2019 (COVID-19) has triggered an unprecedented global health and economic crisis. Early and accurate forecasts of COVID-19 and evaluation of government interventions are crucial for governments to take appropriate interventions to contain the spread of COVID-19. In this work, we propose the Interpretable Temporal Attention Network (ITANet) for COVID-19 forecasting and inferring the importance of government interventions. The proposed model is with an encoder–decoder architecture and employs long short-term memory (LSTM) for temporal feature extraction and multi-head attention for long-term dependency caption. The model simultaneously takes historical information, a priori known future information, and pseudo future information into consideration, where the pseudo future information is learned with the covariate forecasting network (CFN) and multi-task learning (MTL). In addition, we also propose the degraded teacher forcing (DTF) method to train the model efficiently. Compared with other models, the ITANet is more effective in the forecasting of COVID-19 new confirmed cases. The importance of government interventions against COVID-19 is further inferred by the Temporal Covariate Interpreter (TCI) of the model.

## 1. Introduction

Worldwide outbreak of coronavirus disease 2019 (COVID-19) has triggered an unprecedented global health and economic crisis. COVID-19 has caused more than 233 million infections and more than 4.7 million deaths worldwide so far.[1] Early and accurate forecasts of COVID-19 and evaluation of government interventions are crucial for governments to take appropriate interventions and contain the spread of COVID-19.

The forecasting of COVID-19 conducts a prediction of confirmed cases or other indicators caused by COVID-19 in a future horizon. Many works have been proposed to forecast the progression of COVID-19 since its global outbreak.

Compartmental models have been widely used for infectious disease modeling ever since its origin in the early 20th century [1]. In [2], Yang et al. employed modified susceptible–exposed–infected–removed (SEIR) model, taking the move-in and move-out parameters into account, to derive the epidemic trend. In [3], Zhou et al. adjusted the SEIR model for predicting epidemic trends by further considering the contact rate and quarantined proportion of COVID-19 transmission. The major limitations of compartmental models are two-fold: 1) compartmental models only consider limited parameters; 2) compartmental models are unable to incorporate covariates for better forecasting. Recently, researchers have proposed several works to eliminate the limitations of compartmental models by combining them with machine learning algorithms [4,5].

Statistical models, such as Auto-Regressive Integrated Moving Average (ARIMA) and Seasonal Auto-Regressive Integrated Moving Average (SARIMA), were applied to forecast the epidemiological trends of the COVID-19 pandemic in [6–8]. To improve forecasting accuracy, exogenous variables were considered in recent studies for COVID-19 forecasting [9,10]. Statistical models normally rely on multiple assumptions, which limits their applications.

In addition to compartmental models and statistical models, machine learning models and deep learning models have also been widely used in the COVID-19 forecasting problem [11–13]. In [3], the authors proposed a logistic growth model for near-term predictions. In [14], Ardabili et al. investigated several machine learning models, including the multi-layered perceptron and the adaptive network-based fuzzy inference system, to predict the outbreak of COVID-19. As for deep learning models, long short-term memory (LSTM) networks [2,15–17], gated recurrent unit (GRU) networks [16], convolutional neural networks (CNN) [15,17,18], attention-based networks (e.g., Transformer [17,19,20], dot-product attention models [21], graph

[*] Corresponding author.
*E-mail address:* ghyang@jnu.edu.cn (G. Yang).

[1] WHO Coronavirus (COVID-19) Dashboard, as of September 30th, 2021.

https://doi.org/10.1016/j.asoc.2022.108691
1568-4946/© 2022 Elsevier B.V. All rights reserved.

attention networks [22]), etc., were applied to forecast the COVID-19. In addition, hybrid approaches that combine different deep learning methods, such as CNN-LSTM [23], were also investigated for better forecasting of COVID-19. To the best of our knowledge, although machine learning models and deep learning models show better model capacity and flexibility in learning from covariates and forecasting epidemic progression, they suffer from unsatisfying performance of long-term forecasting, overfitting, and poor interpretability.

To overcome the aforementioned limitations, we propose the Interpretable Temporal Attention Network (ITANet) for interpretable multivariate time series forecasting and apply it to the forecasting of COVID-19 epidemic. The model is with an encoder–decoder architecture, where both the encoder and the decoder employ LSTM for extracting temporal features and multi-head attention for capturing long-range dependencies. The model simultaneously takes historical information, a priori known future information, and pseudo future information into consideration. The pseudo future information is learned with the covariate forecasting network (CFN) and multi-task learning (MTL). In addition, We further propose the degraded teacher forcing (DTF) method to train the model efficiently. The main contributions of this work can be summarized as follows:

1. The proposed ITANet has a superior model architecture and learning capacity for time series forecasting and covariate importance extraction;
2. The proposed multi-task learning paradigm has the ability to provide additional supervision information for the model to achieve better forecasting performance.
3. The proposed degraded teacher forcing method is capable of training the model efficiently and mitigating train-test mismatch.
4. The proposed model is capable of providing promising performance in forecasting of the COVID-19, and evaluating the importance of government interventions, which is beneficial for governments to contain the progression of COVID-19.

The remainder of this paper is organized as follows. In Section 2, we introduce the related works of this study. In Section 3, we introduce the architecture and main components of ITANet, as well as the degraded teacher forcing method and the multi-task learning paradigm. In Section 4, we introduce experimental datasets, data preprocessing and experimental settings for comparing our model with existing deep learning models in the forecasting of COVID-19. In Section 5, we show the experimental results of forecasting performance, model uncertainty, model complexity, and ablation studies. We further conduct extensive experiments to demonstrate the benefits of DTF and MTL on the forecasting performance and the benefits of DTF on other models with encoder–decoder architectures. We also evaluate the importance of government interventions with the help of temporal covariate interpreter. Finally, we provide a detailed discussion and conclude this work in Section 6 and Section 7, respectively.

## 2. Related works

### 2.1. COVID-19 forecasting

Multiple types of forecasting models have been applied to forecasting the progression of COVID-19, including compartmental models [2,3], statistical models [6–8], and deep learning models [16,18,20]. However, compartmental models and statistical models suffer from limited expression capability and unsatisfying forecasting performance, while deep learning models are usually blamed for overfitting issues and poor interpretability.

### 2.2. Incorporating and interpreting covariates

To improve forecasting accuracy, researchers have made huge efforts to incorporate covariates so that more information could be utilized. Even for compartmental models and statistical models, recent studies have proposed to incorporate covariates (e.g., changes in the policies [5], number of currently hospitalized patients [10]) to enhance the models' performance. For deep learning models, it would be easier to incorporate covariates (e.g., mobility information [20]). However, these works usually simply combined all incorporated covariates together and inputted them to the networks. Despite some recent studies [24], how to make full use of the covariates and how to interpret the importance of incorporated covariates remain further investigation.

### 2.3. Encoder–decoder models and their training strategies

Encoder–decoder models have been widely employed for time series forecasting in recent years [25,26]. Encoder–decoder models are usually difficult to train, especially when encoder time steps or decoder time steps are long. A well applied training strategy for encoder–decoder models is Teacher Forcing [27], where the decoder has access to all the observed target values at every decoding step to mitigate error propagation. However, since the model cannot always access all the observed target values during the inference stage in a multi-horizon forecasting problem, there inevitably exists a gap between the input information for training and inference, which is the so called train-test mismatch. More efficient training strategies should be proposed to train encoder–decoder models well while mitigating train-test mismatch.

## 3. Proposed method

### 3.1. Interpretable temporal attention network

As shown in Fig. 1, the proposed ITANet is with an encoder–decoder architecture consisting of an encoder for processing historical information and a decoder for processing future information and forecasting. From the very beginning of the encoder and decoder, the input transformation layer transforms the inputs to high-dimensional features, and the temporal covariate interpreter (TCI) layer interprets the importance of each input covariate, respectively. After that, an LSTM layer for extracting temporal features and a multi-head self-attention layer for capturing long-range dependencies are connected to the TCI layer on both the encoder side and decoder side. In addition, the decoder also includes three components: a covariate forecasting network (CFN) for expanding the future inputs, an encoder–decoder attention layer for attending the encoder representation, and a linear layer for final linear regression.

The model takes three types of covariates as inputs, i.e., covariates with historical information only ($x^h$), covariates with historical information and pseudo future information ($x^{pf}$) and covariates with historical information and a priori known future information ($x^{rf}$). The pseudo future information are generated by the covariate forecasting network. Assuming a $\tau_{seq}$-horizon forecasting problem, the original input lengths of $x^h$, $x^{rf}$ and $x^{pf}$ are $\tau_{lag}$, $(\tau_{lag} + \tau_{seq})$ and $\tau_{lag}$, respectively.

#### 3.1.1. Covariate forecasting network

Incorporating more useful covariates and their future information into time series forecasting helps to improve the forecasting

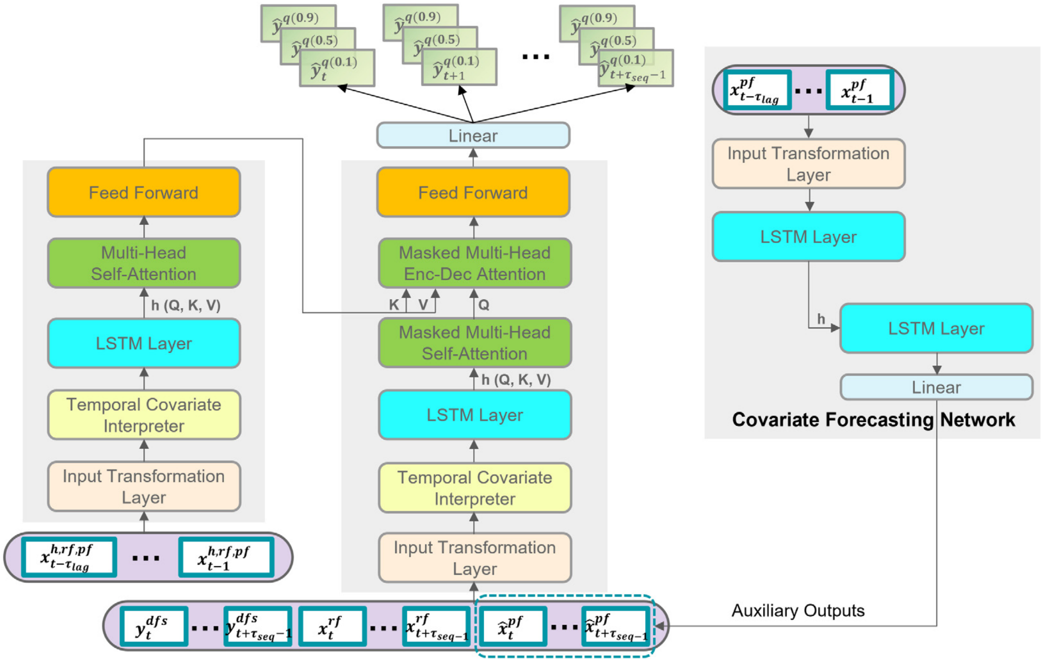

$\hat{y}_t^{q(0.9)}$
$\hat{y}_t^{q(0.5)}$
$\hat{y}_t^{q(0.1)}$
$\hat{y}_{t+1}^{q(0.9)}$
$\hat{y}_{t+1}^{q(0.5)}$
$\hat{y}_{t+1}^{q(0.1)}$
$\hat{y}_{t+\tau_{seq}-1}^{q(0.9)}$
$\hat{y}_{t+\tau_{seq}-1}^{q(0.5)}$
$\hat{y}_{t+\tau_{seq}-1}^{q(0.1)}$
Linear
Feed Forward
Multi-Head Self-Attention
h (Q, K, V)
LSTM Layer
Temporal Covariate Interpreter
Input Transformation Layer
$x_{t-\tau_{lag}}^{h,rf,pf}$
$x_{t-1}^{h,rf,pf}$
Feed Forward
Masked Multi-Head Enc-Dec Attention
K
V
Q
Masked Multi-Head Self-Attention
h (Q, K, V)
LSTM Layer
Temporal Covariate Interpreter
Input Transformation Layer
$y_t^{dfs}$
$y_{t+\tau_{seq}-1}^{dfs}$
$x_t^{rf}$
$x_{t+\tau_{seq}-1}^{rf}$
$\hat{x}_t^{pf}$
$\hat{x}_{t+\tau_{seq}-1}^{pf}$
$x_{t-\tau_{lag}}^{pf}$
$x_{t-1}^{pf}$
Input Transformation Layer
LSTM Layer
h
LSTM Layer
Linear
Covariate Forecasting Network
Auxiliary Outputs

**Fig. 1.** The network architecture of ITANet.

performance of the model. Traditional encoder–decoder architectures usually take only the value of the target sequence at the last time step before the start point of forecasting, i.e., $y_{t-1}$, as the input on the decoder side. In the proposed encoder–decoder architecture, we have incorporated covariates with a priori known future information $x^{rf}$ into the inputs on the decoder side to achieve potential forecasting performance improvement. However, covariates that we have a priori on are limited. Therefore, in this section, we posit that by introducing some critical and predictable covariates to the inputs on the decoder side, the performance of the model can also be promoted. To that end, we propose the covariate forecasting network to generate pseudo future information for some covariates with their historical information and expand the inputs on the decoder side:

$$\hat{x}^{pf}_{t:t+\tau_{seq}} = CFN(x^{pf}_{t-\tau_{lag}:t}). \qquad (1)$$

To avoid overfitting to the historical data, the covariate forecasting network is a lightweight encoder–decoder architecture composed of a single LSTM layer on both the encoder and decoder sides, instead of a complex network architecture.

### 3.1.2. *Input transformation layer*

The covariates considered are either categorical variables or numerical variables. For categorical variables, we successively apply label encoding and linear transformation, to transform each categorical variable to a $d_i^c$-d representation vector. Here the dimension $d_i^c$ is determined by an empirical rule [28]:

$$d_i^c = \min(\text{round}(1.6 * (n_i^c)^{0.56}), \tilde{d}_i^c), \qquad (2)$$

where $n_i^c$ and $\tilde{d}_i^c$ are the number of categories and the predefined maximal embedding size for the $i$th categorical variable, and the round$(\cdot)$ function returns a rounded integer number. Through trial and error, the 1.6 and 0.56 in Eq. (2) empirically give good embedding sizes for categorical variables.

For numerical variables, we simply apply linear transformation to map the original value to a $d^n$-d representation vector. After obtaining the representations for each categorical variable and numerical variable, an additional linear transformation is applied to transform each representation vector into a $h$-d hidden representation space.

### 3.1.3. *Temporal covariate interpreter*

Inspired by the variable selection network in [24], we build the temporal covariate interpreter to interpret the importance of the input covariates. Considering the complexity and expression capability of the network, the temporal covariate interpreter is composed of a 2-layer feed forward network and a softmax activation. Let $\mathbf{h}^m \in \mathbb{R}^h$ denote the transformed representation of the $i$th covariate, covariate-wise importance vector at time step $t$, i.e., $\mathbf{I}_t \in \mathbb{R}^M$, is generated as:

$$\mathbf{I}_t = \text{ELU}\left(\mathbf{h}^F \mathbf{W}_1 + \mathbf{b}_1\right) \mathbf{W}_2 + \mathbf{b}_2, \qquad (3)$$

and

$$\mathbf{h}^F = \text{Flatten}\left([(\mathbf{h}_t^1)^T, \ldots, (\mathbf{h}_t^m)^T \ldots, (\mathbf{h}_t^M)^T]\right), \qquad (4)$$

where ELU is the Exponential Linear Unit activation function [29], $\mathbf{W}_1$, $\mathbf{W}_2$, and $\mathbf{b}_1$, $\mathbf{b}_2$ are learnable weights and biases of the first and second linear layer, respectively. $M$ is the total number of covariates, and the Flatten$(\cdot)$ operation is to generate a flatten vector across all covariates.

After obtaining the covariate-wise importance vector, the output of the TCI layer at time step $t$ can be expressed as:

$$\mathbf{o}_t^{\text{TCI}} = \sum_{m=1}^{M} \mathbf{I}_t^{(m)} \mathbf{h}_t^m, \qquad (5)$$

where $\mathbf{I}_t^{(m)}$ is the $m$th element of $\mathbf{I}_t$.

### 3.1.4. *LSTM layer*

The long short-term memory (LSTM) architecture was originally designed to process long sequential data and remember information for a longer period than the original recurrent neural network (RNN) [30,31]. An LSTM unit has three gates, i.e., forget gate, input gate, and output gate, that control the propagation of two kinds of information, i.e., the cell state and the hidden state. Here we use an LSTM layer to process the representations of inputs to extract temporal features, where the hidden states of all time steps from the LSTM layer are passed to the latter multi-head self-attention layer. The hidden states also provide additional positional information for the attention modules, so that no additional positional encoding is required as inputs to the encoder and decoder.

### 3.1.5. Multi-head attention layers

Although the LSTM layer is able to extract temporal features and provide positional information, the long-range dependencies between the inputs are still difficult to be captured. To solve the problem, attention mechanism is introduced. Let $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ denote the fundamental constituents, i.e., the Query, Key and Value matrices, for calculating the scaled dot-product attention [32], which is specified as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{QK}^T}{\sqrt{d_K}}\right)\mathbf{V}, \tag{6}$$

where $d_K$ is the input dimension.

Multi-head attention, as an extension of the scaled dot-product attention, is more effective and is employed in our model:

$$\text{MultiHead}\,(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}\,(\text{Head}_1, \ldots, \text{Head}_H)\,\mathbf{W}^O, \tag{7}$$

and

$$\text{Head}_{\,i} = \text{Attention}\left(\mathbf{QW}_i^Q, \mathbf{KW}_i^K, \mathbf{VW}_i^V\right), \tag{8}$$

where $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, $\mathbf{W}_i^V$, as well as $\mathbf{W}^O$, are learnable weights which conduct linear transformations from the original representation space to different sub-spaces, and $H$ is the number of attention heads. Multi-head attention is expected to learn from those representation sub-spaces and obtain various features, and thus enhance the model performance.

Following the multi-head attention module, there is also a feed forward network with two linear transformations and one ELU activation in between. Also, a residual connection around each of the multi-head attention module and the feed forward network is employed.

On the encoder side, a multi-head self-attention layer is adopted, where the hidden states from the encoder-side LSTM layer form the $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ matrices. On the decoder side, a multi-head self-attention layer and a encoder–decoder attention layer are adopted. Similarly, the hidden states from the decoder-side LSTM layer form the $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ matrices to the decoder-side multi-head self-attention layer. While for the encoder–decoder attention layer, the $\mathbf{K}$, $\mathbf{V}$ matrices come from the outputs of the encoder-side multi-head self-attention layer and the $\mathbf{Q}$ matrix comes from the outputs of the decoder-side multi-head self-attention layer. It is worth mentioning that to avoid the information leaking from the future steps, masked multi-head attention is employed on the decoder side, which includes an extra look-ahead mask compared to the module on the encoder side.

### 3.1.6. Encoder–decoder architecture

Encoder–decoder architectures have been employed in processing sequential data since the era of recurrent neural networks. Recently, as the development of attention mechanism, the application of encoder–decoder architectures is becoming even broader. For multi-horizon forecasting, the decoder, either of an RNN model or of an attention-based model, normally conducts one-step training and step-by-step inference. During the inference phase, the decoder can only access the last target value before the start point of decoding, therefore the latter decoding steps would be conditioned on the former predictions of the decoder itself. Therefore, the step-by-step inference is not quite effective, especially when the forecasting horizon is long. In this paper, we employ the generative inference proposed in [33] for one-step inference: the decoder is fed with the past $\tau_{seq}$-step target values, and outputs the $\tau_{seq}$-step forecasting results with one decoding step.

In summary, the forecasting results $\hat{y}$ are generated as follows:

$$\hat{y}_{t:t+\tau_{seq}} = Dec\left([x^{rf}_{t:t+\tau_{seq}}, \hat{x}^{pf}_{t:t+\tau_{seq}}, y_{t-\tau_{seq}:t}], C_{t-\tau_{lag}}\right), \tag{9}$$

and

$$C_{t-\tau_{lag}} = Enc\left([x^h_{t-\tau_{lag}:t}, x^{rf}_{t-\tau_{lag}:t}, x^{pf}_{t-\tau_{lag}:t}]\right), \tag{10}$$

where the future part of $x^{pf}$, i.e., $\hat{x}^{pf}_{t:t+\tau_{seq}}$, is given by the covariate forecasting network as Eq. (1) shows.

## 3.2. Degraded teacher forcing

Traditionally, models with encoder–decoder architectures can be trained with Teacher Forcing [27], in which the decoder has access to all the observed target values at every decoding step to mitigate error propagation. However, since the model cannot always access all the observed target values during the inference stage in a multi-horizon forecasting problem, there inevitably exists a gap between the input information for training and inference. In other words, teacher forcing has advantages in properly training the model, but also leads to train-test mismatch. To mitigate train-test mismatch, we propose the degraded teacher forcing for better training the proposed model. Instead of training the model with one-step shifted target sequences as the teacher forcing does, the model is trained with the degraded forcing sequences (DFS). Applying DTF requires 3 types of forcing sequences, i.e., 0-padding sequences, $\tau_{seq}$-step shifted target sequences and one-step shifted target sequences. The Type 1 DFS (0-padding sequences) are to mimic the situations that no target values could be observed. The Type 2 DFS ($\tau_{seq}$-step shifted target sequences) are to mimic the information to the ITANet in the testing stage. The Type 3 DFS (one-step shifted target sequences) are used to force the training so that the model can be properly trained. The combination of the three types of Degraded Forcing Sequences, i.e., the ratios to use the three types of DFS, is further determined by hyper-parameter tuning.

When DTF is applied, the training samples $\{[x_i^h, x_i^{rf}, x_i^{pf}, y_i^{dfs}], y_i\}$ should be generated at the very beginning of each iteration by the sliding window method, as presented in Alg. 1.

For generating validation and testing samples, we simply adopt Type 2 DFS for all samples for one-step inference as we introduced before.

## 3.3. Multi-task learning

As illustrated before, the covariate forecasting network is introduced to generate pseudo future information for some useful covariates. To guarantee the quality of the generated pseudo future information, and to train the forecasting model properly, multi-task learning is introduced to train the whole ITANet with additional supervision from pseudo future known covariates, comparing to the original single-task supervised learning.

Let the task of forecasting COVID-19 confirmed cases be the primary task, and the tasks of forecasting pseudo future known covariates be the auxiliary tasks. The primary task is optimized by minimizing the sum of quantile losses [34] $L_q$ w.r.t multiple quantiles $Q$ and all time points $T$ to forecast:

$$L_{Pri} = \sum_{t\in T}\sum_{q\in Q} L_q(y_t^q, \hat{y}_t^q) \tag{11}$$

, where

$$L_q(y_t^q, \hat{y}_t^q) = \max(q(y_t^q - \hat{y}_t^q),\ 0) + \max((1-q)(\hat{y}_t^q - y_t^q),\ 0). \tag{12}$$

The quantile loss is to offer precise numerical forecasting results and insights of forecast uncertainties by providing multiple forecasting intervals. The commonly used quantiles include: 0.02, 0.1, 0.25, 0.5, 0.75, 0.9, and 0.98.

**Algorithm 1** Method to generate training samples with degraded forcing sequences embedded in.

```
Input:
  $x^h$: covariates with historical information only
  $x^{pf}$: covariates with historical information and pseudo future information
  $x^{rf}$: covariates with historical information and a priori known future information
  $y$: target time series
  $l$: the length of time series
  $\tau_{lag}$: input horizon
  $\tau_{seq}$: forecasting horizon
  $p_1, p_2, p_3$: ratios to use 3 types of DFS
Initialization: $t \leftarrow 0, i \leftarrow 0$
repeat
    $t \leftarrow t+1, i \leftarrow i+1$
    Slide $x^h, x^{pf}, x^{rf}$ from $t$ to $t+\tau_{lag}$, $t+\tau_{lag}$, and $t+\tau_{lag}+\tau_{seq}$, respectively, to generate $x_i^h$, $x_i^{rf}$, and $x_i^{pf}$
    Slide $y$ from $t+\tau_{lag}$ to $t+\tau_{lag}+\tau_{seq}$ to generate $y_i$
    Generate a 0-padding sequence of length $\tau_{seq}$ as a Type 1 DFS $y_i^{DFS1}$
    Slide $y$ from $t+\tau_{lag}-\tau_{seq}$ to $t+\tau_{lag}$ to generate a Type 2 DFS $y_i^{DFS2}$
    Slide $y$ from $t+\tau_{lag}-1$ to $t+\tau_{lag}+\tau_{seq}-1$ to generate a Type 3 DFS $y_i^{DFS3}$
    Form a temporary training sample $([x_i^h, x_i^{rf}, x_i^{pf}, \{y_i^{DFS1}, y_i^{DFS2}, y_i^{DFS3}\}], y_i)$
until $t+(\tau_{lag}+\tau_{seq}) > l$
Randomly select $y_i^{DFS1}$, $y_i^{DFS2}$, and $y_i^{DFS3}$ with the ratios $p_1$, $p_2$ and $p_3$, respectively, to be the label sequence $y_i^{dfs}$ of a training sample
Output:
  training samples $\{[x_i^h, x_i^{rf}, x_i^{pf}, y_i^{dfs}], y_i\}$
```

The auxiliary tasks are optimized by common regression losses (e.g., mean squared error loss, mean absolute error loss, etc.) or common classification losses (e.g., binary cross entropy, categorical cross entropy, etc.) accordingly. For each auxiliary task $a$ across all time points $T$ to forecast, the loss function is determined by:

$$L_{Aux}^a = \begin{cases} \sum_{t \in T} L_{CLS}^a(y_t^a, \hat{y}_t^a), & \text{if } a \text{ is a classification task} \\ \sum_{t \in T} L_{REG}^a(y_t^a, \hat{y}_t^a), & \text{if } a \text{ is a regression task.} \end{cases} \tag{13}$$

The whole ITANet, denoted as $F$, is then jointly optimized as follows:

$$\begin{aligned} F^* &= \mathop{\mathbf{argmin}}_F L \\ &= \mathop{\mathbf{argmin}}_F L_{Pri} + \lambda_a \sum_{a=1}^{N} L_{Aux}^a \end{aligned} \tag{14}$$

, where $N$ is the number of auxiliary tasks, $\lambda_a$ is the weight for balancing auxiliary tasks and the primary task, and $L$ is the joint loss function.

To sum up, the procedures for training the proposed ITANet with degraded teacher forcing and multi-task learning is shown in Alg. 2.

Note that the training set is generated at each iteration with variations, while the validation set is generated beforehand and keep unchanged to accelerate the validation process.

**Algorithm 2** Training the proposed ITANet with degraded teacher forcing and multi-task learning

```
Input:
  $x^h$: covariates with historical information only
  $x^{pf}$: covariates with historical information and pseudo future information
  $x^{rf}$: covariates with historical information and a priori known future information
  $y$: target time series
  $l$: the length of time series
  $\tau_{lag}$: input horizon
  $\tau_{seq}$: forecasting horizon
  $p_1, p_2, p_3$: ratios to use 3 types of DFS
  $V$: validation set
  $Itr$: maximum iteration number
  $\lambda^1, ..., \lambda^a, ..., \lambda^m$: auxiliary task loss coefficients
Initialization: initialize the ITANet, $i \leftarrow 0$
repeat
    $i \leftarrow i+1$
    Run Alg. 1 to get training samples
    Update $F$ with Adam optimizer based on Eq. (14)
until $i > Itr$, or early-stopping is triggered.
Output:
  optimal model $F^*$ with best $L$ on validation set
```

**Table 1**
The exact variables taken in the experiments for three types of inputs.

| | |
|---|---|
| $x^h$ | Historical new confirmed cases, 19 government interventions |
| $x^{pf}$ | Date index, month index, state code |
| $x^{rf}$ | Temperature, air quality index |

## 4. Experiments

### 4.1. Experimental datasets

The variables-of-interest in the experiments, including input variables (i.e., government interventions, and air quality related variables) and target variables (i.e., new confirmed cases), were picked up from the following datasets:

**Confirmed Cases, Confirmed Deaths, and Government Interventions:** Oxford COVID-19 Government Response Tracker [35] is a tool to rigorously and consistently track policy responses around the world. Government interventions, including pharmaceutical interventions and non-pharmaceutical interventions, are divided into four categories: containment and closure policies, economic policies, health system policies and miscellaneous policies. The dataset provides daily updated government interventions, confirmed cases and deaths from more than 180 countries and regions. The dataset also provides state-level data for some countries including the United States. Note that miscellaneous policies are excluded since they are basically described with free text.

**Temperature and Air Quality Index:** The temperature and air quality index data from the World Air Quality Index project. The dataset provides the values for each air pollutant species (e.g, PM10, PM2.5) and meteorological data (e.g., wind, precipitation, temperature), covering about 380 major cities in the world.

Specifically, the exact variables taken in the experiments for three types of inputs (i.e., $x^h$ (variables with historical information only), $x^{pf}$ (variables with historical information and pseudo future information), $x^{rf}$ (variables with historical information and a priori known future information)) are shown in Table 1:

**Table 2**
Hyper-parameters, corresponding tuning spaces, and best hyper-parameter settings of the ITANet for the three states.

| Hyper-parameter | Tuning Space | Best for CA | Best for IL | Best for TX |
|---|---|---|---|---|
| $d^n$ | 4, 8 | 4 | 4 | 4 |
| $h$ | 4, 8, 16, 32, 64 | 16 | 32 | 16 |
| $H$ | 1, 2, 4, 8, 16, 32, 64 | 8 | 16 | 16 |
| $p_{drop}$ | 0.1, 0.2, 0.3, 0.4, 0.5 | 0.5 | 0.5 | 0.3 |
| $(p_1, p_2, p_3)$ | (0.1, 0.1, 0.8), (0.15, 0.15, 0.7), (0.25, 0.25, 0.5) | (0.15, 0.15, 0.7) | (0.15, 0.15, 0.7) | (0.25, 0.25, 0.5) |

$d^n$: mapped vector dimension of numerical variables;
$h$: dimension of hidden representations (including combined hidden representation, hidden states of LSTM layers in the CFN or the main encoder–decoder network) and feed-forward size (inner dimension of feed forward networks within the TCI or following multi-head attention modules);
$H$: number of attention heads;
$p_{drop}$: dropout rate;
$(p_1, p_2, p_3)$: ratios to use 3 types of DFS.

### 4.2. Data preprocessing

The data extracted from above datasets cannot be directly used for model training or analysis.

Missing values and abnormal values are common in raw data, and thus further data filling and cleaning is required. For numerical variables, we filled the blanks with 0. For categorical variables, we filled the blanks with forward filling method, which forwarded the last observed value to the blanks till the next observed value, considering that these categorical variables may last for a few days. We checked the codebook of the government intervention dataset to make sure there was no abnormal values in these variables.

On considering the data availability, we conducted the experiments on 3 states of the United States: Illinois, California and Texas. The time series between April 1st, 2020 and April 28th, 2021 was chosen and split into three parts:

- training set, from April 1st, 2020 to March 31st, 2021;
- validation set, from April 1st, 2021 to April 14th, 2021;
- testing set, from April 15th, 2021 to April 28th, 2021.

Sliding window method was applied to generate training, validation and testing samples as Alg. 1 shows. Since the available data points (in days) were limited, we slid the window by a stride=1.

To avoid the training being dominated by some large scale numerical variables, pre-scaling was conducted on all numerical variables. The pre-scaler for each variable was generated only with the data in the training set. The pre-scaling was done by Z-score normalization, where population mean $\bar{x}$ and population standard deviation $\sigma$ were firstly calculated, and then the original data is transformed according to the following equation:

$$x' = \frac{(x - \bar{x})}{\sigma}. \tag{15}$$

### 4.3. Experimental settings

We train the model for 14-horizon forecasting with 28-horizon inputs under the multi-task learning paradigm and the joint loss was calculated as Eq. (14). Here we used quantiles including: 0.1, 0.5, and 0.9. Due to data availability, the auxiliary tasks include forecasting the future temperature and Air Quality Index (AQI). We set $\lambda_a = 0.1$ for all the auxiliary tasks.

Mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE) were introduced as the performance evaluation metrics:

$$MAE = \frac{1}{\tau_{seq}} \sum_{t=1}^{\tau_{seq}} \left| y_t - \hat{y}_t \right|, \tag{16}$$

$$RMSE = \sqrt{\frac{1}{\tau_{seq}} \sum_{t=1}^{\tau_{seq}} (y_t - \hat{y}_t)^2}, \tag{17}$$

**Table 3**
Forecasting performance of ITANet and baseline models.

| Model | State | MAE | RMSE | MAPE |
|---|---|---|---|---|
| CNN | California | 1068.15 | 1273.53 | 0.5808 |
| | Illinois | 790.64 | 900.45 | 0.3233 |
| | Texas | 2704.02 | 2932.47 | 1.1087 |
| LSTM | California | 3986.68 | 4170.14 | 2.5184 |
| | Illinois | 809.13 | 924.78 | 0.3135 |
| | Texas | 1179.96 | 1584.90 | 0.6166 |
| Transformer | California | 658.54 | 891.26 | 0.5413 |
| | Illinois | 848.7 | 983.08 | 0.2862 |
| | Texas | 1595.42 | 2049.38 | 0.4353 |
| TFT | California | 993.01 | 1191.83 | 0.7656 |
| | Illinois | 694.77 | 840.82 | 0.2274 |
| | Texas | 1282.84 | 1664.10 | 0.6281 |
| ITANet | California | **597.69** | **697.44** | **0.3950** |
| | Illinois | **294.46** | **392.49** | **0.1215** |
| | Texas | **875.44** | **1073.59** | **0.3462** |

$$MAPE = \frac{1}{\tau_{seq}} \sum_{t=1}^{\tau_{seq}} \left| \frac{y_t - \hat{y}_t}{y_t} \right| \tag{18}$$

, where $y_t$ and $\hat{y}_t$ denote the true value and predicted value of new conformed cases at time t, and $\tau_{seq}$ is the length of time horizons to forecast.

We compared the proposed ITANet with the other four deep learning models, including CNN [18], LSTM [16], Transformer [20] and Temporal Fusion Transformer (TFT) [24]. As naive compartmental methods and statistical methods cannot incorporate covariates, they were not considered in comparison.

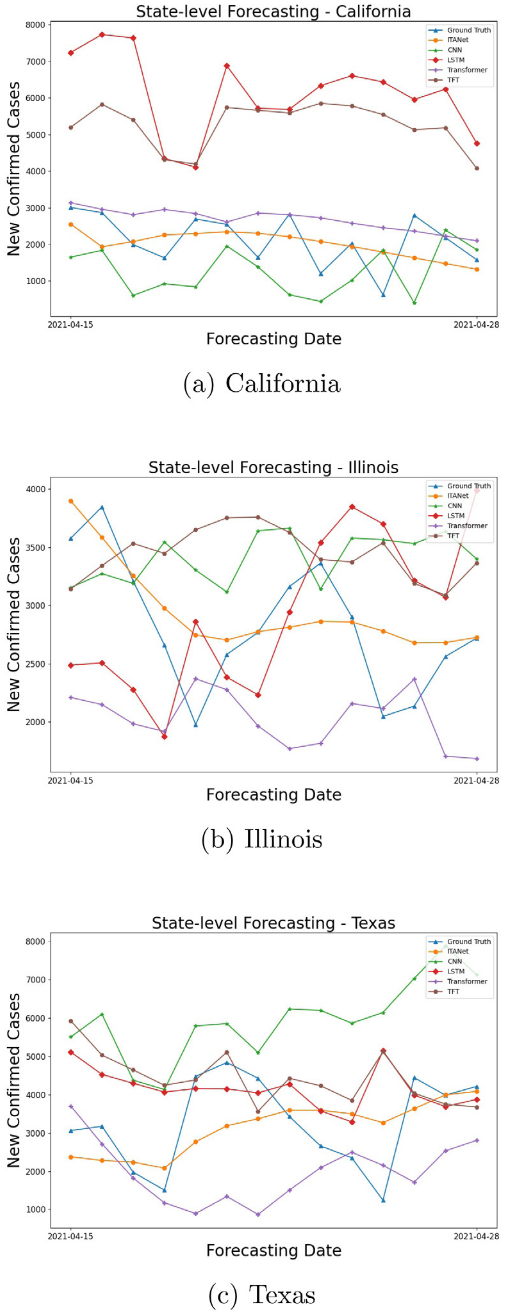
The models were implemented, trained and tested in Python with TensorFlow 1.15. The models were trained over 100 epochs and the batch size was set to 256. The models were trained and tested on a HPC with 2 Intel(R) Xeon(R) Gold 6230 CPUs (@2.10 GHz), 256 GB RAM, and 2 NVIDIA V100 32 GB GPUs.

Grid search was employed to determine the hyper-parameters of the neural networks. The hyper-parameter tuning spaces for the proposed model are listed in Table 2. The model with minimum validation loss was selected as the best model to conduct the following evaluations and comparisons on the testing set.

## 5. Experimental results

### 5.1. Performance evaluation

We compared the proposed ITANet with the other four baseline models, i.e., CNN, LSTM, Transformer and TFT, evaluated on the forecasting of the COVID-19 new confirmed cases of 3 states of the United States, i.e., California, Illinois and Texas. Experimental results are shown in Table 3 and Fig. 2.

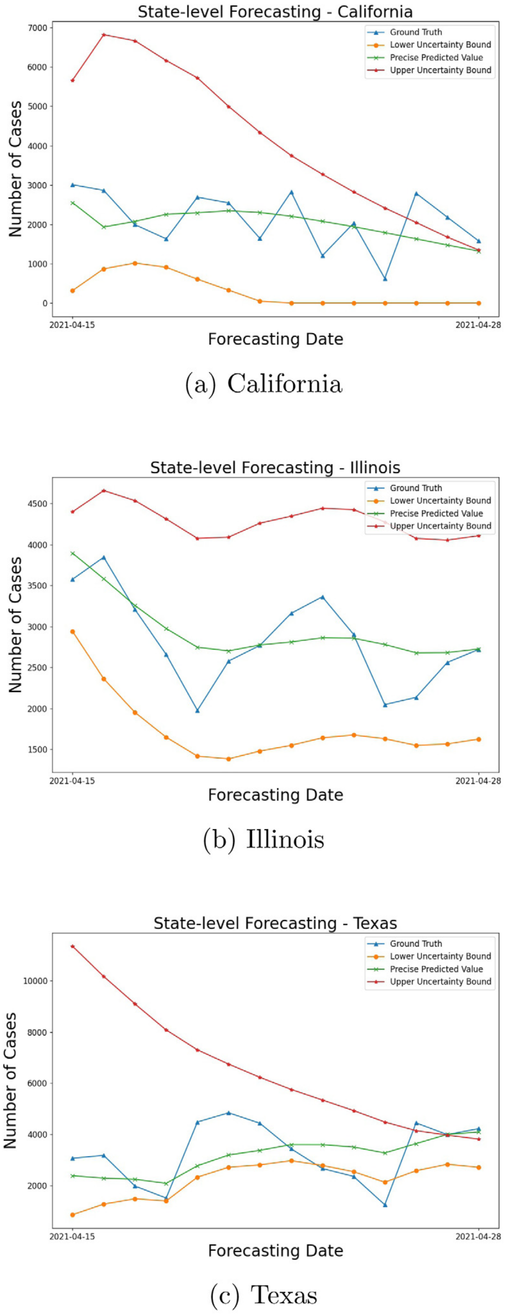


(a) California

(b) Illinois

(c) Texas

**Fig. 2.** Examples of COVID-19 forecasting results from Apr. 15 to Apr. 28 by ITANet and other models.



(a) California

(b) Illinois

(c) Texas

**Fig. 3.** Prediction intervals for COVID-19 forecasting by ITANet in California, Illinois and Texas.

The ITANet outperforms all the other models in terms of all evaluation metrics, i.e., MAE, RMSE and MAPE. From Fig. 2, we can find that the proposed model is better than the others not only in numerical regression, but also in trend prediction.

### 5.2. Model uncertainty

In order to study model uncertainty, we trained the proposed ITANet with quantile loss as defined in Eq. (11), so that the models provided prediction intervals in addition to precise value predictions. Since the model was trained with 0.1, 0.5, 0.9 quantiles, we use 0.1 quantile prediction and 0.9 quantile prediction as the lower bound and upper bound of the prediction intervals, respectively. The prediction intervals of new confirmed cases for 14-horizon forecasting are shown in Fig. 3.

From Fig. 3, we can find that the prediction intervals given by our model can accurately reflect the maximal and minimal scales of the progression of the COVID-19 epidemic. The prediction intervals can help governments to be fully prepared for the epidemic, which is of great significance.

### 5.3. Model complexity

To compare the complexity of the proposed model with the baseline models, the total number of parameters and floating-point operations (FLOPs) of each model are listed in Table 4. It can be seen from Table 4 that with the well designed network

**Table 4**
Model complexity of ITANet and baseline models.

| Model | State | No. of Params | FLOPs |
|---|---|---|---|
| CNN | California<br>Illinois<br>Texas | 146.010k<br>90.936k<br>149.482k | 291.476k<br>181.332k<br>298.228k |
| LSTM | California<br>Illinois<br>Texas | 53.488k<br>53.664k<br>390.666k | 106.986k<br>107.338k<br>189.184k |
| Transformer | California<br>Illinois<br>Texas | 48.368k<br>48.544k<br>48.384k | 97.583k<br>97.935k<br>97.675k |
| TFT | California<br>Illinois<br>Texas | 58.896k<br>123.062k<br>786.432k | 114.970k<br>274.924k<br>1610.574k |
| ITANet | California<br>Illinois<br>Texas | **32.230k**<br>**32.406k**<br>**32.438k** | **70.652k**<br>**71.004k**<br>**71.212k** |

**Table 5**
Ablation studies for multi-task learning and degraded teacher forcing.

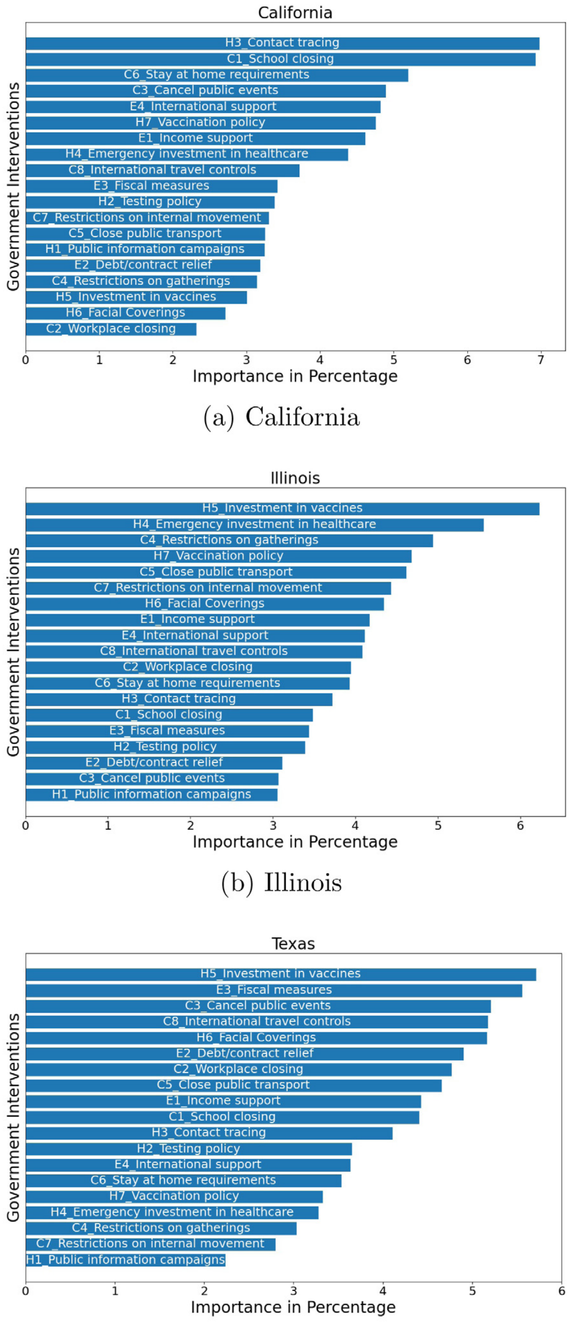
| Model | State | MAE | RMSE | MAPE |
|---|---|---|---|---|
| ITANet | California<br>Illinois<br>Texas | **597.69**<br>**294.46**<br>**875.44** | **697.44**<br>**392.49**<br>**1073.59** | **0.3950**<br>**0.1215**<br>**0.3462** |
| ITANet w/o MTL | California<br>Illinois<br>Texas | 1097.59<br>521.14<br>1313.54 | 1351.52<br>598.91<br>1621.78 | 0.6971<br>0.1758<br>0.4554 |
| ITANet w/o DTF | California<br>Illinois<br>Texas | 944.75<br>863.37<br>1206.08 | 1101.76<br>985.63<br>1367.08 | 0.5077<br>0.3573<br>0.4676 |
| ITANet w/o MTL&DTF | California<br>Illinois<br>Texas | 1663.99<br>858.09<br>1913.67 | 1903.21<br>957.54<br>2230.83 | 0.7683<br>0.3497<br>0.5367 |

architecture, the proposed ITANet is very efficient in exploiting input information and achieves the best performance with the least computational complexity.

### 5.4. Ablation studies

In this work, we proposed a covariate forecasting network with multi-task learning (MTL) to provide additional supervision for training a model with better forecasting performance. In addition, we also proposed the degraded teacher forcing (DTF) method to train the model efficiently while reducing the train-test mismatch. In this section, we conducted ablation studies to investigate the advantages of multi-task learning and degraded teacher forcing. For comparison, three models, i.e., ITANet without MTL, ITANet without DTF, and ITANet without MTL & DTF, were evaluated. Table 5 shows the performance of the three models in forecasting COVID-19 new confirmed cases in California, Illinois and Texas under the aforementioned configurations.

From Table 5, we can find that both MTL and DTF have benefits to the COVID-19 forecasting. Without MTL or DTF, the performance of the model drops significantly with a few exceptions. For training without DTF, the performance loss may due to that the forecasting horizons are too long. The accumulated training errors make the model difficult to train. When DTF is applied, the model is trained with the guidance of the degraded forcing sequences to mitigate the training errors. At the same time, due to our elaborate design of the degraded forcing sequences and the training procedure, the DTF also avoids great train-test mismatch while training the model efficiently. As for the studies for MTL, we posited that to provide highly relevant future information may help the forecasting, so the performance loss may due to the reduction of supervision information or network parameters.



(a) California

(b) Illinois

(c) Texas

**Fig. 4.** The importance of government interventions against COVID-19 in California, Illinois and Texas.

From Table 5, we can also find that DTF is more useful than MTL. The possible reason was that we only provided future temperature and AQI information for supervision considering data availability, which may have limited contributions to the outputs. However, the key idea behind this method provides a direction for the use of future information for additional supervision by combining the covariate forecasting network and multi-task learning. Given sufficient useful covariates, the method is expected to contribute more.

**Table 6**
Effectiveness of degraded teacher forcing for other models with encoder–decoder architectures.

| Model | State | MAE | RMSE | MAPE |
|---|---|---|---|---|
| LSTM | California | 3986.68 | 4170.14 | 2.5184 |
| | Illinois | 809.13 | 924.78 | **0.3135** |
| | Texas | 1179.96 | 1584.90 | 0.6166 |
| LSTM w/ DTF | California | **886.51** | **1128.62** | **0.6780** |
| | Illinois | **796.26** | **911.11** | 0.3202 |
| | Texas | **917.86** | **1304.03** | **0.3110** |
| Transformer | California | **658.54** | 891.26 | 0.5413 |
| | Illinois | 848.70 | 983.08 | 0.2862 |
| | Texas | 1595.42 | 2049.38 | 0.4353 |
| Transformer w/ DTF | California | 677.26 | **821.80** | **0.5342** |
| | Illinois | **460.62** | **552.17** | **0.1585** |
| | Texas | **921.79** | **1133.11** | **0.4028** |
| TFT | California | 993.01 | 1191.83 | 0.7656 |
| | Illinois | 694.77 | 840.82 | 0.2274 |
| | Texas | **1282.84** | 1664.10 | 0.6281 |
| TFT w/ DTF | California | **1426.35** | **1653.07** | **0.7120** |
| | Illinois | **525.92** | **645.01** | **0.2209** |
| | Texas | 1342.25 | **1563.98** | **0.5842** |

Since DTF is also suitable for other models with encoder–decoder architectures, such as TFT, LSTM and Transformer, we also compared the differences in forecasting performance between models trained with and without DTF. The results in Table 6 demonstrate that the proposed DTF training method is effective and can achieve a consistent performance improvement on the models with encoder–decoder architectures.

### 5.5. Covariate importance

One of the motivations of this work is to interpret covariate importance in time series forecasting. As for the forecasting of COVID-19, we hope to determine which interventions are most important and thus provide a guidance for future epidemic containment. We consider 3 categories of government interventions, i.e., containment and closure policies, health system policies, and economic policies, in our experiments, which are denoted by 'Ca_b', 'Ha_b', 'Ea_b', where 'a' is the order of a policy within each category of interventions, and 'b' is the name of the policy. Fig. 4 shows the importance of each government intervention regarding the three states obtained by the temporal covariate interpreter, where the importance is measured by importance in percentage (i.e., the sum of the importance of all covariates is 100 percentages). Note that the importance of those non-policy covariates is not shown in Fig. 4.

To better evaluate the importance of government interventions, we scored three categories of interventions, i.e., containment and closure policies, economic policies, health system policies, according to their importance within each category, and summed the scores obtained in three states as the final importance score of each intervention. The scores of all the government interventions are shown in Table 7.

From Table 7, we can find that international travel controls, canceling public events and closing public transport are among the top 3 important containment and closure policies. Investment in vaccines, vaccination policy and contact tracing are considered as the most important health system policies. For economic policies, the scores are not consistent for different states, but may imply that income support and debt/contract relief might be the most and least important measures, respectively. It is worth mentioning that the importance of a government intervention can be affected by a lot of factors, such as when the intervention starts, how stringent it is, how long it lasts, etc, resulting in the differences of the importance of government interventions between the states. We expect that these findings can provide valuable suggestions for the governments to choose more effective control measures to contain the progression of the COVID-19.

## 6. Discussion

In this work, we propose a novel forecasting model, i.e., ITANet, as well as a set of methods to achieve better COVID-19 forecasting performance. According to the experimental results, the proposed ITANet outperforms the other baseline models in 14-day COVID-19 forecasting for three US states. More importantly, the proposed model is able to reveal the importance of covariates including government interventions through the temporal covariate interpreter. This capability provides an evaluation method on the government interventions in containing the progression of COVID-19.

The proposed model makes full use of various kinds of information, including historical information, a priori known future information, and pseudo future information learned with the covariate forecasting network and multi-task learning. The proposed degraded teacher forcing method is able to train the model efficiently.

It is worth mentioning that the proposed model is applicable for other time series forecasting problems, in addition to the forecasting of COVID-19 confirmed cases, which may provide satisfying performance of long-term forecasting and interpretable covariate importance. The DTF method is also applicable for training other models with encoder–decoder architectures.

One limitation of this work is that although the proposed ITANet achieves the best performance in forecasting COVID-19 confirmed cases, it is not able to always keep satisfying forecasting performance at each time step within the 14-day forecasting horizons. This may be due to the lack of training data or the use of sliding window method to cut the original time series, resulting in insufficient feature capture. The model can be further improved to support longer-term COVID-19 forecasting by designing better attention mechanisms.

Another limitation is that although the proposed model is able to interpret the covariate importance at each forecasting horizon, the actual effectiveness of a government intervention is not provided. It is complicate to determine the actual effectiveness of a government intervention, since it can be affected by a lot of factors (e.g., duration, strictness of the intervention). It is our future direction to fully determine the effectiveness of these interventions.

## 7. Conclusion

In this work, we proposed a method based on ITANet, a novel deep neural network, to conduct COVID-19 forecasting and infer the importance of government interventions. We used the covariate forecasting network and multi-task learning paradigm to introduce more supervision information for training the model to improve forecasting performance. We further proposed the degraded teacher forcing method to train the model efficiently while mitigating train-test mismatch. The ITANet was compared with other deep learning models, including CNN, LSTM, Transformer and Temporal Fusion Transformer. The experimental results demonstrated the effectiveness of our proposed model in the forecasting of COVID-19.

**Table 7**
The importance ranking scores of government interventions.

| Intervention Category | Interventions | California | Illinois | Texas | Total Score |
|---|---|---|---|---|---|
| Containment and closure policies | C1_School closing | 8 | 2 | 4 | 14 |
| | C2_Workplace closing | 1 | 4 | 6 | 11 |
| | C3_Cancel public events | 6 | 1 | 8 | **15** |
| | C4_Restrictions on gatherings | 2 | 8 | 2 | 12 |
| | C5_Close public transport | 3 | 7 | 5 | **15** |
| | C6_Stay at home requirements | 7 | 3 | 3 | 13 |
| | C7_Restrictions on internal movement | 4 | 6 | 1 | 11 |
| | C8_International travel controls | 5 | 5 | 7 | **17** |
| Health system policies | H1_Public information campaigns | 3 | 1 | 1 | 5 |
| | H2_Testing policy | 4 | 2 | 4 | 10 |
| | H3_Contact tracing | 7 | 3 | 5 | **15** |
| | H4_Emergency investment in healthcare | 5 | 6 | 2 | 13 |
| | H5_Investment in vaccines | 2 | 7 | 7 | **16** |
| | H6_Facial Coverings | 1 | 4 | 6 | 11 |
| | H7_Vaccination Policy | 6 | 5 | 3 | **14** |
| Economic policies | E1_Income support | 3 | 4 | 2 | **9** |
| | E2_Debt/contract relief | 1 | 1 | 3 | 5 |
| | E3_Fiscal measures | 2 | 2 | 4 | **8** |
| | E4_International support | 4 | 3 | 1 | **8** |

## CRediT authorship contribution statement

**Binggui Zhou:** Data curation, Methodology, Software, Formal analysis, Writing – original draft. **Guanghua Yang:** Conceptualization, Supervision, Writing – review & editing. **Zheng Shi:** Validation, Writing – review & editing. **Shaodan Ma:** Supervision, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This work was supported in part by the Guangdong-Macau Joint Funding Project under Grant 2021A0505080015, and in part by the National Key Research and Development Program of China under Grant 2017YFE0120600.

## References

[1] R. Ross, An application of the theory of probabilities to the study of a priori pathometry.—Part I, Proc. R. Soc. London. Ser. A Contain. Pap. A Math. Phys. Charact. 92 (638) (1916) 204–230, http://dx.doi.org/10.1098/rspa.1916.0007.

[2] Z. Yang, Z. Zeng, K. Wang, S.-S. Wong, W. Liang, M. Zanin, P. Liu, X. Cao, Z. Gao, Z. Mai, J. Liang, X. Liu, S. Li, Y. Li, F. Ye, W. Guan, Y. Yang, F. Li, S. Luo, Y. Xie, B. Liu, Z. Wang, S. Zhang, Y. Wang, N. Zhong, J. He, Modified SEIR and AI prediction of the epidemics trend of COVID-19 in China under public health interventions, J. Thoracic Disease 12 (3) (2020) http://dx.doi.org/10.21037/jtd.2020.02.64.

[3] X. Zhou, X. Ma, N. Hong, L. Su, Y. Ma, J. He, H. Jiang, C. Liu, G. Shan, W. Zhu, S. Zhang, Y. Long, Forecasting the worldwide spread of COVID-19 based on logistic model and SEIR model, 2020, MedRxiv, 2020.03.26.20044289, Cold Spring Harbor Laboratory Press, http://dx.doi.org/10.1101/2020.03.26.20044289.

[4] F. Amaral, W. Casaca, C.M. Oishi, J.A. Cuminato, Towards providing effective data-driven responses to predict the Covid-19 in são Paulo and Brazil, Sensors 21 (2) (2021) 540, http://dx.doi.org/10.3390/s21020540.

[5] R. Vega, L. Flores, R. Greiner, SIMLR: Machine learning inside the SIR model for COVID-19 forecasting, Forecasting 4 (1) (2022) 72–94, http://dx.doi.org/10.3390/forecast4010005.

[6] H. Alabdulrazzaq, M.N. Alenezi, Y. Rawajfih, B.A. Alghannam, A.A. Al-Hassan, F.S. Al-Anzi, On the accuracy of ARIMA based prediction of COVID-19 spread, Results Phys. 27 (2021) 104509, http://dx.doi.org/10.1016/j.rinp.2021.104509.

[7] K.E. ArunKumar, D.V. Kalaga, C.M. Sai Kumar, G. Chilkoor, M. Kawaji, T.M. Brenza, Forecasting the dynamics of cumulative COVID-19 cases (confirmed, recovered and deaths) for top-16 countries using statistical machine learning models: Auto-regressive integrated moving average (ARIMA) and seasonal auto-regressive integrated moving average (SARIMA), Appl. Soft Comput. 103 (2021) 107161, http://dx.doi.org/10.1016/j.asoc.2021.107161.

[8] P. Kumar, H. Kalita, S. Patairiya, Y.D. Sharma, C. Nanda, M. Rani, J. Rahmani, A.S. Bhagavathula, Forecasting the dynamics of COVID-19 pandemic in top 15 countries in April 2020: ARIMA model with machine learning approach, 2020, MedRxiv, 2020.03.30.20046227, Cold Spring Harbor Laboratory Press, http://dx.doi.org/10.1101/2020.03.30.20046227.

[9] B.S. Aji, Indwiarti, A.A. Rohmawati, Forecasting number of COVID-19 cases in Indonesia with ARIMA and ARIMAX models, in: 2021 9th International Conference on Information and Communication Technology, ICoICT, 2021, pp. 71–75, http://dx.doi.org/10.1109/ICoICT52021.2021.9527453.

[10] M. Toutiaee, X. Li, Y. Chaudhari, S. Sivaraja, A. Venkataraj, I. Javeri, Y. Ke, I. Arpinar, N. Lazar, J. Miller, Improving COVID-19 forecasting using exogenous variables, 2021, [Cs, Stat] arXiv:2107.10397.

[11] N. Anjum, M. Asif Kiran, F. Jabeen, Z. Yang, C. Huang, S. Noor, K. Imran, I. Ali, E.M. Mohamed, Intelligent COVID-19 forecasting, diagnoses and monitoring systems: A survey, 2021, TechRxiv, http://dx.doi.org/10.36227/techrxiv.15172488.v1.

[12] A. Majeed, S.O. Hwang, Data-driven analytics leveraging artificial intelligence in the era of COVID-19: An insightful review of recent developments, Symmetry 14 (1) (2022) 16, http://dx.doi.org/10.3390/sym14010016.

[13] G.R. Shinde, A.B. Kalamkar, P.N. Mahalle, N. Dey, J. Chaki, A.E. Hassanien, Forecasting models for coronavirus disease (COVID-19): A survey of the state-of-the-art, SN Comput. Sci. 1 (4) (2020) 197, http://dx.doi.org/10.1007/s42979-020-00209-9.

[14] S.F. Ardabili, A. Mosavi, P. Ghamisi, F. Ferdinand, A.R. Varkonyi-Koczy, U. Reuter, T. Rabczuk, P.M. Atkinson, COVID-19 outbreak prediction with machine learning, Algorithms 13 (10) (2020) 249, http://dx.doi.org/10.3390/a13100249.

[15] K.N. Nabi, M.T. Tahmid, A. Rafi, M.E. Kader, M.A. Haider, Forecasting COVID-19 cases: A comparative analysis between recurrent and convolutional neural networks, Results Phys. 24 (2021) 104137, http://dx.doi.org/10.1016/j.rinp.2021.104137.

[16] F. Shahid, A. Zameer, M. Muneeb, Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM, Chaos, Solitons, Fractals 140 (2020) 110212, http://dx.doi.org/10.1016/j.chaos.2020.110212.

[17] H. Abbasimehr, R. Paki, Prediction of COVID-19 confirmed cases combining deep learning methods and Bayesian optimization, Chaos, Solitons, Fractals 142 (2021) 110511, http://dx.doi.org/10.1016/j.chaos.2020.110511.

[18] C.-J. Huang, Y.-H. Chen, Y. Ma, P.-H. Kuo, Multiple-input deep convolutional neural network model for COVID-19 forecasting in China, 2020, MedRxiv, 2020.03.23.20041608, Cold Spring Harbor Laboratory Press, http://dx.doi.org/10.1101/2020.03.23.20041608.

[19] R. Wang, D. Maddix, C. Faloutsos, Y. Wang, R. Yu, Bridging physics-based and data-driven modeling for learning dynamical systems, in: Proceedings of the 3rd Conference on Learning for Dynamics and Control, PMLR, 2021, pp. 385–398.

[20] S. Er, S. Yang, T. Zhao, County aggregation mixup AuGmEntation (COURAGE) COVID-19 prediction, Sci. Rep. 11 (1) (2021) 14262, http://dx.doi.org/10.1038/s41598-021-93545-6.

[21] X. Jin, Y.-X. Wang, X. Yan, Inter-series attention model for COVID-19 forecasting, in: Proceedings of the 2021 SIAM International Conference on Data Mining, SDM, in: Proceedings, Society for Industrial and Applied Mathematics, 2021, pp. 495–503, http://dx.doi.org/10.1137/1.9781611976700.56.

[22] J. Gao, R. Sharma, C. Qian, L.M. Glass, J. Spaeder, J. Romberg, J. Sun, C. Xiao, STAN: Spatio-temporal attention network for pandemic prediction using real-world evidence, J. Am. Med. Inf. Assoc. 28 (4) (2021) 733–743, http://dx.doi.org/10.1093/jamia/ocaa322.

[23] Z.M. Zain, N.M. Alturki, COVID-19 pandemic forecasting using CNN-LSTM: A hybrid approach, J. Control Sci. Eng. 2021 (2021) e8785636, http://dx.doi.org/10.1155/2021/8785636.

[24] B. Lim, S.Ö. Arık, N. Loeff, T. Pfister, Temporal fusion transformers for interpretable multi-horizon time series forecasting, Int. J. Forecast. 37 (4) (2021) 1748–1764, http://dx.doi.org/10.1016/j.ijforecast.2021.03.012.

[25] S. Du, T. Li, Y. Yang, S.-J. Horng, Multivariate time series forecasting via attention-based encoder–decoder framework, Neurocomputing 388 (2020) 269–279, http://dx.doi.org/10.1016/j.neucom.2019.12.118.

[26] N. Wu, B. Green, X. Ben, S. O'Banion, Deep transformer models for time series forecasting: The influenza prevalence case, 2020, [Cs, Stat] arXiv:2001.08317.

[27] R.J. Williams, D. Zipser, A learning algorithm for continually running fully recurrent neural networks, Neural Comput. 1 (2) (1989) 270–280, http://dx.doi.org/10.1162/neco.1989.1.2.270.

[28] J. Howard, S. Gugger, Fastai: A layered API for deep learning, Information 11 (2) (2020) 108, http://dx.doi.org/10.3390/info11020108.

[29] D.-A. Clevert, T. Unterthiner, S. Hochreiter, Fast and accurate deep network learning by exponential linear units (ELUs), 2016, [Cs] arXiv:1511.07289.

[30] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780, http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[31] F.A. Gers, J. Schmidhuber, F. Cummins, Learning to forget: Continual prediction with LSTM, Neural Comput. 12 (10) (2000) 2451–2471, http://dx.doi.org/10.1162/089976600300015015.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, 2017, [Cs] arXiv:1706.03762.

[33] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, W. Zhang, Informer: Beyond efficient transformer for long sequence time-series forecasting, 2021, [Cs] arXiv:2012.07436.

[34] R. Wen, K. Torkkola, B. Narayanaswamy, D. Madeka, A multi-horizon quantile recurrent forecaster, 2018, [Stat] arXiv:1711.11053.

[35] T. Hale, N. Angrist, R. Goldszmidt, B. Kira, A. Petherick, T. Phillips, S. Webster, E. Cameron-Blake, L. Hallas, S. Majumdar, H. Tatlow, A global panel database of pandemic policies (Oxford COVID-19 government response tracker), Nat. Hum. Behav. (2021) 1–10, http://dx.doi.org/10.1038/s41562-021-01079-8.